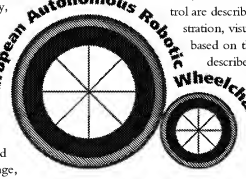
A FRIEND for Assisting Handicapped People

The Semiautonomous Robotic System "FRIEND" Consists of an Electric Wheelchair with a Robotic Arm and Utilizes a Speech Interface

by CHRISTIAN MARTENS, NILS RUCHEL, OLIVER LANG, OLEG IVLEV, and AXEL GRÄSER

People with upper-limb impairments, including people with multiple sclerosis, spasticity, cerebral palsy, paraplegia, or stroke, depend on a personal assistant for daily life situations and in the working environment. The robot arm MANUS was designed for such people, but they frequently cannot use it because of their disability. The control elements require flexibility and coordination in the hand that they do not have. To facilitate access to technological aids, control functionality of the aids has to be improved. Additionally, it should be possible to enter more abstract commands to simplify the use of the system. This has been partly realized in the commercially available HANDY system. The HANDY user can choose between five fixed actions, such as eating different kinds of food from a tray or being served a drink. However, the restricted flexibility of HANDY is a big disadvantage, as a fixed environment is a prerequisite.



To place the full capacity of technical systems like the robot arm MANUS with 6 degrees of freedom (DOF) at the user's disposal, a shared control structure is necessary. Low-level commands and taking advantage of all cognitive abilities of the user lead to full control flexibility. To relieve the user, semiautonomous sensor-based actions are also included in the control structure. Actions, such as gripping an object in an unstructured environment or pouring a cup with a drink and serving it to the disabled user, may be started by simple commands. The user is still responsible for decisions in situations where errors are possible, such as locating objects or planning a sequence of preprogrammed actions.

This article presents the robotic system FRIEND, which was developed at the University of Bremen's Institute of Automation (IAT). This system offers increased control functionality for disabled users. FRIEND consists of an electric wheelchair, equipped with the robot arm MANUS. Both devices are controlled by a computer. The man-machine interface (MMI) consists on a flat screen and a speech interface. FRIEND's hardware and software are described first. The current state of development is then presented, as well as research results that will be integrated soon. After a short explanation of the speech interface, the methods developed for semiautonomous control are described. These are programming by demonstration, visual servoing, and configuration planning based on the method of imaginary links. We also describe the state of integration and our experience to date.

Hardware and Software

The robotic system consists of an electric wheelchair (Meyra, Germany), a MANUS robot arm (Exact Dynamics, Holland), and a Dual Pentium PC, which is mounted in a special rigid box behind the wheelchair (Fig. 1). The robot arm is linked to the PC via a CAN-bus to exchange commands and measurement data. The wheelchair receives commands via an RS232 interface. A tray mounted on the left side of the wheelchair and a flat LCD display as part of the MMI complete the design.

To integrate semiautonomous actions, the system includes cameras. Currently, a mini camera is mounted on top of the gripper. For future applications, a stereo-system will be placed behind the user's head.

Figure 2 shows the current software layout, as well as anticipated software components. Currently, the speech interface is implemented as an MMI [1], but the system architecture facil-

1070-9932/01/$10.00©2001IEEE

itates the addition of other input elements like chin-control, suck-blow-control, and control via eye movements.

The user commands FRIEND by short naturally spoken words, and the MMI transforms them into strings using a speech recognition software. If a valid command is given, the interpreter activates the necessary software component to perform the task demanded. Types of commands include direct control commands for the arm, the start command for a semiautonomous action, and commands to activate a complete sequence.

Preprogrammed actions, like pouring, may be generated off-line, using either a classical teach-in procedure or by robot programming by demonstration (RPD). RPD combines programmed movements with scripts that can be parameterized. A disadvantage of preprogrammed movements is that no variations are admissible. The item (the cup or bottle), it's position, and initial conditions (the fluid level) must not be different compared to the programming situation. RPD may partially eliminate this disadvantage.

In an unstructured environment the use of preprogrammed actions is unsuitable. Therefore, FRIEND uses a combination of control by the user and autonomous control by the system. If, for instance, the user wants something to drink, he first will have to move the robot arm close to a bottle using speech commands like "Arm left" or "Arm up." As soon as the hand-mounted camera recognizes the object, the user initiates a pick action with the command "Pick!" In the pick action, the gripper is moved into an adequate position relative to the bottle using visual servoing, and the bottle is gripped automatically.

An additional complication occurs if a desired object isn't located in the robot's workspace. A routine work problem is to remove a folder from a shelf. In this case, the wheelchair first has to move close to the shelf. When the wheelchair is in a suitable position, close enough to the shelf to reach the folder, the camera system has to recognize the folder. During this recognition process, the user supports the system with verbal commands to move the cameras close to the folder. When the folder is detected successfully, the command "Dock!" activates the docking action. A suitable wheelchair position is determined with the help of the imaginary links method to solve the inverse kinematic problem. In this case the wheelchair and the robot arm form a redundant system with 9 DOF. During every robot arm and wheelchair action, an intelligent part of the arm controller, the so-called Kinematic Configuration Controller (KCC) [2], computes collision-free trajectories with additional sensor information.

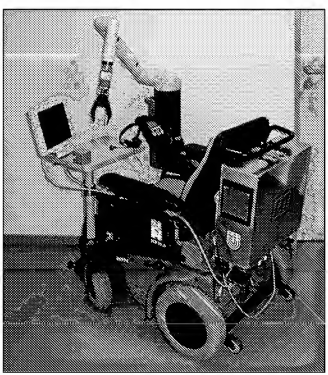


Fig. 1. *The robotic system FRIEND.*

Speech Control

The speech processing system translates naturally spoken words into commands. It consist of the following modules:

- Speech recognizer: Translates naturally spoken words into strings.
- Command interpreter: Interprets the strings as machine commands.

The speech recognizer consists of the speech recognition software ViaVoice Gold (by IBM), which translates naturally spoken commands entered via a microphone into specific words using predetermined grammar. The recognized words are transmitted to the command interpreter, which translates the words or text sequences received into system commands. For safety reasons, the set of commands is organized into a hierarchical command tree. To suppress the misinterpretation of commands, such as interpreted noise or misspelled words, a path in the tree must be completed to cause a system action. This minimizes the possibility of erroneous interpretations. If the system is in direct command mode and the command interpreter doesn't receive a permissible command within a fixed period of time, the system automatically returns to a safe state. The entire tree and the current state of the command are represented graphically on the flat screen, because the user can easily recognize the state of the system if it is presented in the form of pictograms (Fig. 3).

If the interpreter receives a permissible word, this word is highlighted in the tree with a gray background. In addition, all recognized words are shown in the left display box. This makes it easy for the user to monitor the system's behavior and react quickly if an error occurs.

Currently, four controlling modes of the robot arm are implemented. These are

1. Control world coordinates.
2. Control joint coordinates.
3. Control tool coordinates.
4. Activate action sequences.

It is possible to switch between different coordinate systems or to activate the semiautonomous actions mode and se-

quences via the spoken command "Change Mode." This makes the system very flexible.

Programming Complex Motions by Demonstration

To program new motions, the robotic system is equipped with two programming environments. In keyboard mode, the robot is programmed in the traditional way, available in almost all robots. In this mode, with the keyboard point-to-point positions on a trajectory are generated that are traced and stored in a database. In programming by demonstration mode (RPD), the programmer demonstrates the task to be executed with his own hand. The motions are measured, recorded, and processed so that the robot can reproduce them. Many approaches described in the literature [3–5] share a common feature: they are designed mainly for simple pick-and-place applications like those found in industry, such as loading palettes and sorting and feeding parts. Neither the demonstrated motion trajectory nor the dynamics of the motion, such as the speed or general time response, are considered. But in the field of rehabilitation robots like FRIEND, where the tasks are much more complicated, this information is of great impor-

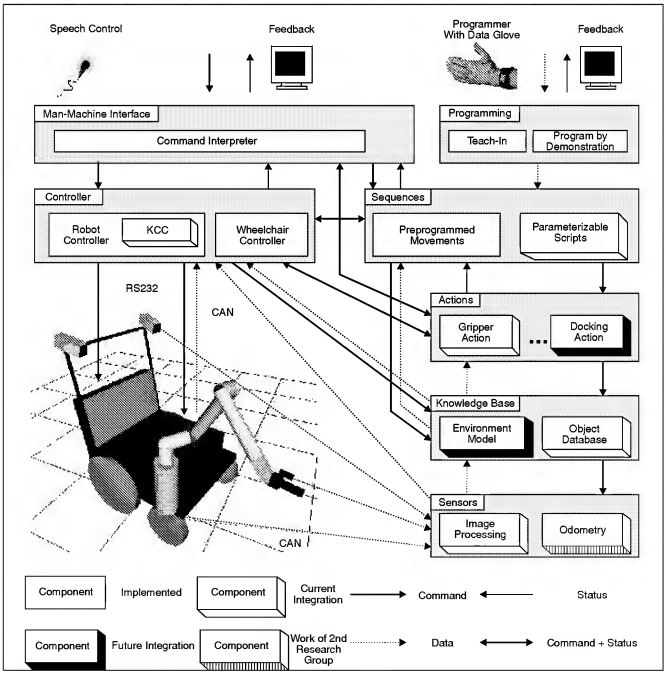


Fig. 2. Architecture of the system FRIEND.

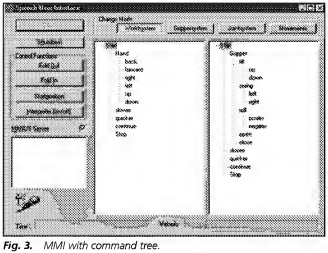


Fig. 3. MMI with command tree.

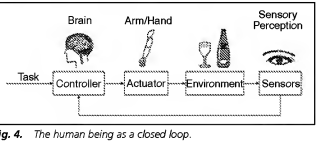


Fig. 4. The human being as a closed loop.

tance to enable the robot to execute these tasks correctly. Moreover, the methods mentioned don't consider that the human operator acts as a part of a closed loop (Fig. 4), acting as a sensor, control algorithm, and actuator. The loop is closed across the environment [6].

This closed loop enables a human being to execute even very complex trajectories without difficulty and in spite of variable initial conditions. For example, in the described scenario "pouring a glass from a bottle," the human continually observes the level of the glass and the flow from the bottle visually. With this information, he controls the motion of the bottle and fills the glass to a constant level with different shaped bottles and glasses, independent of the initial levels. To achieve similar robustness with a robot, a feedback loop must be installed.

The demonstrated motions are divided into subsequences. The robot may repeat some as demonstrated, whereas others have to be part of the control loop. Additional sensor information is included and the controller will modify the recorded trajectory. In the next section we explain this method in detail for the "pouring a glass" scenario.

Trajectory Generation

First, the trajectory for the pouring process has to be generated (Fig. 5) To detect the motion of the bottle, a 6 DOF position sensor attached to the bottle and a transmitter, which represents the reference system for the motion detection, is used.

The programmer grasps an open bottle, places it in a relative pose to a glass, starts recording, and demonstrates the characteristic motion of this action. During the demonstration, the current pose of the bottle relative to the transmitter is sampled at regular time intervals. These pose data and the corresponding time represent the motion trajectory and are stored in an ordered list. With this detection method it is possible to precisely copy the demonstrated motion and its dynamics. The investigation of the pouring process yielded that it is sufficient to demonstrate one process, in which a full bottle is emptied completely. This overall trajectory contains all the movements for all initial levels.

Transformation Relative to a Specific Reference System

To enable the robot to repeat the demonstrated task, the data recorded relative to the transmitter has first to be transformed to another reference system. This might be the robot base. For service-robot applications, however, it is useful to consider movements of gripped objects relative to other objects. In our example, this may be the movement of the bottleneck relative to the glass rim. The benefit of this approach is that the demonstrated motion can be performed relative to the glass rim in different places. Moreover, the motion becomes independent of the specific features of the objects used during the demonstration, because the object information is shifted into homogeneous transformation matrices. Hence, the motion can easily be transferred to different shapes of bottles and glasses. In the execution phase the motion of the robot has to be calculated from the movement of the bottleneck in relation to the glass. In the present implementation of FRIEND, the necessary object information and the position of the glass relative to the robot are assumed to be known. In the future, a stereo camera system will be used to measure the distances online.

Division into Open-Loop and Closed-Loop Subsequences

Extremely simplified, the pouring process can be divided into three subsequences. These subsequences, sent and executed sequentially by the robot, are:

1. Sensor-observed, open-loop start subsequence: From the starting pose, the overall trajectory is followed until flow from the bottle starts.

2. Closed-loop controlled pouring subsequence: As soon as liquid flows from the bottle, the glass is filled using a closed-loop control. The automatic controller has to keep the flow at a desired rate until the desired level or volume in the glass is reached.

3. Open-loop closing subsequence: When the desired level or volume is detected, the robot switches to another demonstrated trajectory to avoid dripping.

To implement these subsequences, the transformed trajectory is stored in an ordered list and a pointer defines the next set point value for the robot. Only in open-loop mode may the speed be modified by changing the sampling time or pointer increment. In closed loop, the speed and trajectory are modified by the controller. The controller does not cal-

culate a new pose but a pointer in the ordered list, which defines a pose.

This procedure has been implemented and tested with the rehabilitation robot MANUS. The actual level of the fluid in the glass is determined with a camera (Fig. 6) and the flow from the bottle and the volume in the glass are calculated. The controller translates the difference between the desired and the computed flow to new offsets in the trajectory list and controls the flow in this way. Figure 7 shows the volume in the glass, the computed flow from the bottle and the control output.

Robust Positioning of the Gripper

Grasping an object with a support system like FRIEND in direct control mode is a time-consuming task for the user, even with the speech input device. On the other hand, the implementation of a general grasp utility leads to very complex algorithms. Since the number of frequently handled objects in our scenario is limited, a supporting utility that handles most of the necessary objects is already very helpful.

FRIEND uses a grasp utility for known labeled objects. Visual servoing with a gripper-mounted camera is used in conjunction with teaching by showing (TbS). With TbS the gripper and the labeled object are placed in the desired relative position, the corresponding image is taken, and the desired features (as image features we use the centers of gravity of the four marker points) are extracted and stored.

If the same object has to be grasped again, it is necessary to return the gripper to the taught grasp position. This can be done either manually using basic spoken commands or automatically using visual servoing with the gripper-mounted camera. In the latter case, the user has only to move the gripper into a position where the camera can detect the label on the object and to initiate the pick action (Fig. 8, left). To support the user, a live camera image is represented on the flat screen. With visual servoing, the gripper is moved to the position where the actual image features correspond to the taught features (Fig. 8, right).

The visual servoing controller computes the change in the position of the robot by comparing the measured and the desired image features and moves the robot to the calculated position. In the new position, another image is taken and the control algorithm is repeated until the taught relative position is reached. The robot is driven in Cartesian space with a look-and-move strategy.

A controller suitable in a rehabilitation robotic system must:

- drive the gripper from any reasonable starting position to the gripping position and
- must ensure that the object marker remains in the image during the motion.

Due to the slow sampling time of the vision system in cluttered environments, a small number of steps are important also. In order to compare the behavior of the different automatic controllers systematically, a computer simulation was developed, named *multi-position test* [7].

Traditional Image-Based Automatic Controllers

The literature usually discusses automatic controllers with a linear system model. The image Jacobian is constant or adapted by parameter estimation at the operating point. In every sampling step, a robot movement $\mathbf{b}$ is computed—using the image Jacobian and the current image error—to minimize the image error $\mathbf{e}$. Finally the manipulated variable $\mathbf{u}$ is computed using an amplification factor $k \leq 1$:

$$\mathbf{u} = k \cdot \mathbf{b} \tag{1}$$

Traditionally, such automatic controllers work with a constant amplification factor k. But at start up of visual control, when the image error is quite large, an automatic controller with a constant gain would compute a large output. To guarantee safe op-

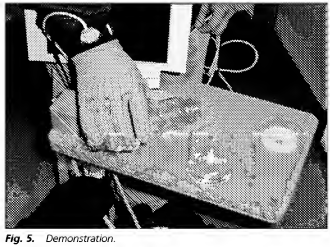


Fig. 5. *Demonstration.*

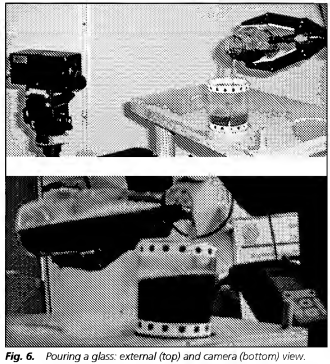


Fig. 6. *Pouring a glass: external (top) and camera (bottom) view.*

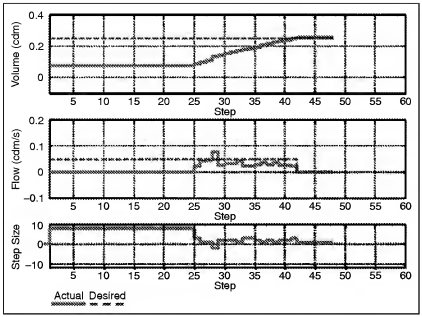


Fig. 7. Experimental results.

eration in the whole workspace in spite of the inaccuracy of a linear system model, a small value of k has to be chosen. However, this necessitates a high number of sampling steps.

Trust Region Method-Based Automatic Controllers

To reduce the number of necessary control steps, we developed a controller that limits the model-predicted movement of the label in the image according to a constant parameter α. This leads to an adapted amplification factor k^*:

$$k^* = \max\left\{1, \frac{\text{maximum movement in the image}}{\text{model} - \text{predicted movement in the image}}\right\} \tag{2}$$

With k^*, an automatic controller that limits the movement in the image at start up is achieved. Close to the target position, when the image-based error is small, the limitation becomes inactive (i.e., $k^* = 1$) and fast convergence is obtained. A value of $\alpha = 0.08$ mm, which guarantees convergence in the entire workspace, was determined experimentally. The number of necessary sampling steps is significantly reduced compared to the traditional controllers.

α may also be adapted in every control step depending on the difference between the model and reality by introducing a *trust region*. The resulting adaptive controller improves the control result [8]. Jägersand [9] has already applied *trust region* methods for visual servoing applications. He controlled the joint coordinates of the robot and used primarily a stationary stereo camera system. Since the 1960s, trust region methods have been used to minimize nonlinear functions in optimization theory. They are also called *restricted step methods*.

The controller with adapted α has been tested with an industrial robot and has been implemented in FRIEND recently. The approach is calibration-robust and hence suitable for the roughly calibrated miniature camera at MANUS [10].

One of the main problems for visual servoing with a gripper-mounted camera is the variable object size resulting from camera movement. This leads to very high demands on the object recognition algorithms. We developed an approach using a zoom camera to obtain constant object sizes during movement [11].

Unfortunately, zoom cameras are currently too heavy and too bulky to be mounted on the MANUS arm, but they can be mounted behind the user, with a pan-tilt system for visual servoing of the robotic arm. For this purpose the method of virtual points [12] has been developed at the IAT and will be implemented in FRIEND soon.

(Semi-)Automatic Docking-Action

Abating the complexity, the wheelchair has to be stationary in the room before the grip action is launched. The object to be grasped has to be within the working space of the robot. Otherwise, the wheelchair must be moved closer to the object. The location of the wheelchair (i.e., Cartesian position X_g and Y_g on the floor and orientation angle Ψ_g) has to be selected in a way that the object is in the workspace of the manipulator and can be grasped without colliding with potential obstacles. We call this process the "docking action."

The obvious strategy of "the closer the better" is not always correct for a room with obstacles, as illustrated in Fig. 9 and in Fig. 10. When the wheelchair is too close to the object, a collision with the shelves results. Choosing a suitable loca-

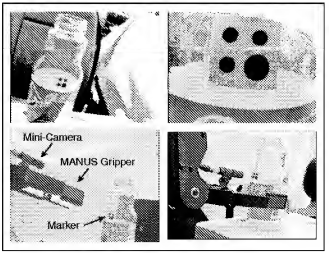


Fig. 8. Start (left) and end (right) of pick action.

tion in a real environment manually is very arduous, since this location can only be determined during the grasping process. For users with limited head mobility, it is almost impossible to estimate a clustered spatial situation correctly.

A (semi-)autonomous docking system assists the user. A suitable wheelchair location is calculated from spatial information about the object and obstacles. To recognize the object and reconstruct the scene, stereo cameras are needed.

To automatically determine a suitable docking location, the inverse kinematic problem for the entire robotic system with 9 DOF has to be solved. This means, from the kinematic equation,

$$ {}_{s}\mathbf{g} = \boldsymbol{f}({}_{3}\mathbf{B},\ {}_{6}\mathbf{A}) \tag{3} $$

the wheelchair coordinates ${}_{3}\mathbf{B} = [X_B\, Y_B\, \Psi_B]^T$ and the manipulator configurations, ${}_{6}\mathbf{A} = [A_1, \ldots, A_6]^T$ have to be calculated. ${}_{s}\mathbf{g}$ is an s-dimensional vector of the desired Cartesian location of the gripper, and the nonlinear vector function $\boldsymbol{f}$ specifies the kinematic structure of the robotic system.

The dimension s of the vector ${}_{s}\mathbf{g}$ depends on the object to be gripped. When the gripper location is determined completely (i.e., both the Cartesian coordinates X_T, Y_T, Z_T of the *Tool Center Point* and the gripper orientation *Yaw, Pitch,* and *Roll* are unequivocally defined), the vector ${}_{s}\mathbf{g} = [X_T, Y_T, Z_T, \textit{Yaw, Pitch, Roll}]$ has a size $s = 6$. Grasping a book may serve as an example. However, such a fully defined specification occurs in service robotics only rarely. Figure 11 shows three typical grasping cases. If a cylinder (e.g., a glass or a bottle) must be grasped, the orientation angle *Yaw* is not defined precisely (it can have an arbitrary value between 0 and ±180°). This leads to ${}_{s}\mathbf{g} = [X_T, Y_T, Z_T, \textit{Pitch, Roll}]$ and $s = 5$. When grasping a ball-shaped object such as an apple or an egg, all three orientation angles are free, ${}_{s}\mathbf{g} = [X_T, Y_T, Z_T]$ and $s = 3$.

This means that equation system (3) is an underdetermined system with a high degree of redundancy $r = 10 - s \geq 4$. Using the conventional method of inverse kinematic analysis for redundant manipulators, the values of redundant degrees of freedom have to be specified in addition to the vector ${}_{s}\mathbf{g}$. This yields a so-called functional-closed solution, which contains joint variables as parameters. To determine a suitable robot configuration, all possible combinations of redundant joint values have to be considered. For a robotic system with high redundancy, as in our case, this kind of solution has a huge computational cost.

To calculate a suitable robot configuration in strict real time, an explicit closed solution of the inverse kinematics problem is desirable. This solution can be obtained using the method of imaginary links [13]. In this approach, the redundancy problem is solved by complementing equation system (3) with the corresponding number $r = 9 - s$ of additional scalar equations of the type

$$ \lambda_k^2 = (\mathbf{R}_k - \mathbf{R}_{Ak})^{\mathrm{T}} \cdot (\mathbf{R}_k - \mathbf{R}_{Ak}). \tag{4} $$

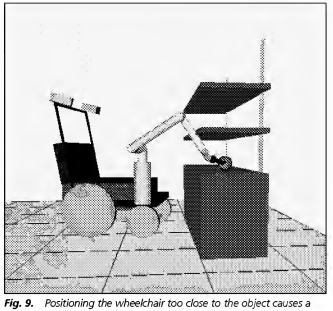


Fig. 9. Positioning the wheelchair too close to the object causes a collision with the shelf.

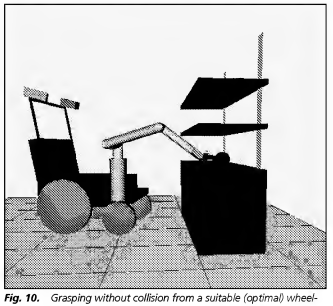


Fig. 10. Grasping without collision from a suitable (optimal) wheelchair location.

These equations define the arrangement of redundant joints in Cartesian space with respect to specific points $\mathbf{R}_{Ak}$.

The simple form of these additional equations makes it possible to derive a closed analytical solution for robots with regular geometry; i.e., those with perpendicular or parallel joint axes:

$$ \begin{bmatrix} {}_{3}\mathbf{B} \\ {}_{6}\mathbf{A} \end{bmatrix} = \mathbf{i}_e({}_{s}\mathbf{g}, \Lambda, \Re). \tag{5} $$

In addition to the desired gripper location $\mathbf{g}$, the extended inverse function $\mathbf{i}_e$ now contains a set of independent parameters $\Lambda = \{\lambda_i\}$ and $\Re = \{\mathbf{R}_{Ai}\}$. The parameters λ_i can be viewed as the lengths of imaginary links with nonactuated

spherical joints at either end. These links connect the redundant joints with virtual anchor points, placed at Cartesian points $\mathbf{R}_{Ai}$. By choosing the parameters for the imaginary links, information about the free workspace can be introduced directly into the solution. The local free space around an arbitrary Cartesian point $\mathbf{R}_{Ai}$ can be approximated using elastic spheres, called *bubbles* (Fig. 12), with variable radii λ_i. When $\lambda_i < \lambda_{max}$, the redundant joints are guaranteed to be in collision-free positions.

Figure 13 shows the corresponding collision-free configurations for planar motion of the robotic system FRIEND with Y_B constant, $\Psi_B = 0$, and $A_1 = A_4 = 0$. The symbolic inverse functions for this case can be found in [14].

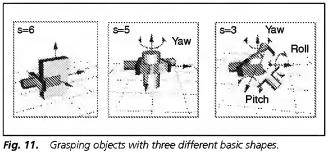


Fig. 11. *Grasping objects with three different basic shapes.*

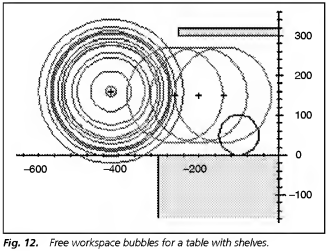


Fig. 12. *Free workspace bubbles for a table with shelves.*

Fig. 13. *Configurations without collisions for planar movement of the robot system FRIEND.*

In this example, the distance between the wheelchair and the table must be not less than 30 cm, in order to grasp the object without a collision.

The equations result in eight possible solutions for each pair of free workspace bubbles for elbow and wrist joints. One out of 8 × (*number of bubble pairs*) configurations has to be chosen for the automatic docking task. This selection can invoke different criteria, such as the maximum distance of robot links to obstacles and maximum manipulability. Due to the use of the closed solution in Eq. (5), the configuration generation and optimization processes can be conducted in parallel. This increases the real-time capacity of the proposed method.

Experiences and Future Development

After a short time of training the user is able to grip different kinds of objects and move them close to him. But some parts of these manipulation actions are very laborious; especially exact positioning, as they are needed to grasp an object, and the execution of a complex task, like pouring water from a bottle into a glass, are hard to handle. The reasons are time delays in the speech recognition system and the tremendous amount of elementary commands that are needed to realize a complex task. Particularly, the last point isn't justifiable because of the long command sentences that had been introduced for security reasons. It requires the whole concentration of the user and is very tiresome. It is possible to shorten the command sentences, but in practice in a noisy environment, experience showed that these long sentences were necessary. As discussed, a marked improvement arises with the introduction of semiautonomous actions. In combination with the speech input system, this provides the user with a set of robust actions, which can be combined to accomplish complex tasks.

In spite of the theoretical and practical results presented above, a huge amount of research and development is still required. Therefore, future work will focus on the analysis and implementation of semiautonomous actions, the simplification of the MMI, and improvement of the safety functions.

Keywords

Rehabilitation robots, speech-control interface, redundant robots, skill-based autonomous control, programming by demonstration.

References

[1] B. Borgerding, O. Ivlev, C. Martens, N. Ruchel, and A. Gräser, "FRIEND - Functional robot arm with user friendly interface for disabled people," in *Proc. 5th European Conf. for the Advancement of Assistive Technology*. Düsseldorf, Nov. 1-4, 1999, to be published.

[2] S. Alfi , O. Ivlev, C. Martens, and A. Gräser, "Simulation tool for kinematic configuration control technology for dexterous robots. A total networked industry environment," in *Proc. IECON'99*, San Jose, CA, USA, Nov. 29 - Dec. 3, 1999, accepted for publication.

[3] Y. Jiar, M. Wheeler, and K. Ikeuchi, "Hand action perception for robot programming," in *Proc. 1996 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 3, pp. 1586-1593.

[4] Y. Kuniyoshi, I. Murayuki, and H. Inoue, "Learning by watching: Reusable task knowledge from visual observation of human performance," *IEEE Trans. Robot. Automat*, vol. 10, pp. 799-822, 1994.

[5] H. Friedrich, S. Münch, and R. Dillmann, "Robot programming by demonstration (RPD): Supporting the induction by human interaction," *Machine Learning*, vol. 23, no. 2-3, pp. 163-89, 1996.

[6] N. Ruchel, O. Lang, and A. Gräser: "Service-robot programming by demonstration with sensor integration," in *Proc. 1999 IEEE Hong Kong Symp. on Robotics and Control*, vol. 1, pp. 226-231.

[7] R. Vogel: "Visuelle Regelung eines Roboters mit sechs Freiheitsgraden unter Berücksichtigung von Objektgröße und Objektposition im Bild," Universität Bremen, Diplomarbeit, Juni 1999 (in German).

[8] N.T. Siebel, O. Lang, F. Wirth, and A. Gräser: "Robuste Positionierung eines Roboters mittels Visual Servoing unter Verwendung einer Trust-Region-Methode," at the Jahrestagung der Deutschen Forschungsvereinigung für Meß-, Regelungs- und Systemtechnik (DFMRS), 11-12. November 1999, Bremen (in German).

[9] M. Jägersand: "Visual servoing using Trust Region Methods and estimation of the full coupled visualm-otor jacobian," in *Proc. IASTED Applications of Control and Robotics Conf.*, 1996, pp. 105-108.

[10] O. Lang, C. Martens, and A. Gräser: "Realisation of a semiautonomous gripping-skill for the support of disabled people," in *Proc. Robotik 2000*, Jun. 29-30, 2000, Berlin.

[11] O. Lang and A. Gräser: "Visual control of 6 DOF robots with constant object size in the image by means of zoom camera," presented at IECON'99, the 25th Annual Conference of the IEEE Industrial Electronics Society, San Jose, CA, Nov. 29 - Dec. 3, 1999.

[12] T. Trittin and A. Gräser: "The method of virtual points for autonomous, image-based robot control," presented at 14th World Congress of International Federation of Automatic Control, Beijing, July 5-9, 1999, Preprints vol. B, pp. 293-298.

[13] O. Ivlev and A. Gräser, "An analytical method for the inverse kinematics of redundant robots," in *Proc. 3rd ECPD Int. Conf. on Advanced Robots, Intelligent Automation and Active Systems*, 1997, pp. 416-421.

[14] O. Ivlev and A. Gräser, "Resolving redundancy of series kinematic chains through imaginary links," in *Proc. CESA'98 IMACS Multiconference. Computational Engineering in Systems Applications*, Tunisia, 1998, pp. 477-482.

[15] O. Ivlev and A.Gräser, "Explicit symbolic solution of the inverse kinematics for redundant robotic systems," *at-Automatisierungstechnik*, 47, 11, 1999, pp. 523-531 (in German).

Christian Martens received his B.S. degree in technical computer sciences in 1994 from the Flensburg University of Applied Sciences and his M.S. degree in electrical engineering from the University of Hagen, Germany, in 2000. During 1994–1995 he worked for a vehicle simulation company as a software engineer. Between 1995-2000 he worked as a technical assistant at the Institute of Automation, University of Bremen. In this position, he has been the technical project manager of the rehabilitation robotic system FRIEND for the past four years. Currently, he is a Ph.D. student at the IAT with research interests in agent-oriented software frameworks and intelligent robotic planning systems.

Oliver Lang received his M.S. degree in electrical engineering from the University of Bremen in 1995. During 1995 to 1999, he was a research assistant at the Institute of Automation at the same university. He worked in the field of service robotics, visual servoing, and zoom camera calibration, and he wrote his doctoral thesis (Ph.D.) about these subjects. During 1999–2000 he was with the Department of Computer Science at the University of Groningen, The Netherlands, as a coordinator for industrial automation, where he lectured about real-time systems. Since September 2000 he has been working as a software engineer in a company that is developing coordinate measurement machines.

Nils Ruchel received his M.S. in electrical engineering from the University of Bremen in 1995. From 1995 to 2000 he was a research assistant at the Institute of Automation at the same university. He worked in the field of new robotic programming methods (programming by demonstration), where he is writing his Ph.D. thesis. Currently, he is working as a project manager in a company that is developing fully automated assembly lines for car motors.

Oleg Ivlev received his M.S. degree in electrical engineering and control science in 1978, and the Ph.D. degree in dynamic and control in 1986, both from the State University of Dnepropetrovsk, Ukraine. From 1977 he joined the Ukrainian National Academy of Sciences, where he held several academic and managing positions, conducting research in the area of mathematical modeling and control of dynamic systems. Concurrently he served as an adjunct professor at the State University of Dnepropetrovsk, teaching industrial robotic systems. During 1995–1997 he was a visiting scientist at the Institute of Automation, University of Bremen, Germany. Currently, he has been leading the project *Kinematic Control of Redundant Robotic Systems* at the same institute, supported by DFG-German Research Foundation.

Axel Gräser is a full professor and head of the Department of Process Automation and Robotics at the Institute of Automation, University of Bremen. Prior to this position he was a professor of automation at the Koblenz University of Applied Sciences. He received the B.S. degree in communications engineering in 1972, the M.S. degree in electrical engineering from the Technical University Karlsruhe in 1976, and the Ph.D. degree in electrical engineering from the Technical University Darmstadt 1982. From 1982 to 1990 he held several industrial positions, heading the development of automation systems and real-time software applications for the pulp and paper industry as well as the chemical and manufacturing industry. His current research areas are service and rehabilitation robotics, complete control of product quality, as well as measurement and control technology to automate logistic processes.

Address for Correspondence: Christian Martens, Institute of Automation (IAT), University of Bremen, Kufsteiner Str. NW 1, 28359 Bremen, Germany.